# roslaunch

# 목차

1. Intro to roslaunch
2. XML Syntax
3. How to use roslaunch
4. TimedRoslaunch

# 01

Intro to roslaunch

# 01. Intro to roslaunch

- **Roslaunch**

- roslaunch는 여러 개의 node를 실행시킬 수 있다. (rosrun을 여러 번 사용할 필요가 없다)
- roslaunch 실행 시에는 ros master가 실행되어 있지 않아도 없을 경우, 자동으로 생성이 된다.
- roslaunch는 같은 네트워크 상에 있는 다른 PC에 있는 것을 실행하는 것도 가능하다.
- Parameter 서버에 Parameter를 등록함과 동시에 실행하는 것이 쉽다.
- roslaunch는 다른 roslaunch 파일을 재사용해서 생성할 수 있다.
- $ roslaunch package_name file.launch 명령어를 통해 사용할 수 있다.
- roslaunch 파일은 XML 문법을 따른다.

# 02

XML Syntax

# 02. XML Syntax

- **XML은 Extensible Markup Language의 줄임말이다.**
- 실제 문법은 HTML과 유사하지만, 정해져있는 태그 외에도 원하는 태그를 골라서 사용할 수 있다.
- <Hello>와 같은 형태로 여는 태그를 사용하며, </Hello>와 같은 형태로 닫는 태그를 사용할 수 있다.
- XML 자체는 HTML과 달리 내용 전달에 사용하는 태그의 제한이 없고, 전달하려는 내용에 대해서 적어주면 충분하다.
- 일반적으로 특정 데이터의 내용을 전달 시에, 데이터 전달의 표준으로 사용하기 위해 개발되었다.

# 02. XML Syntax

- <?xml version=”1.0” encoding=”euc-kr” ?>
- <수업 과목>
- <수학>
- <학생 명단>
- <이름>Rachael</이름>
- <학년>3</학년>
- <나이>15</나이>
- </학생 명단>
- </수학>
- </수업 과목>

# 03

How to use roslaunch

# 03. How to use roslaunch

- roslaunch의 경우 여러 개의 node를 쉽게 실행할 수 있으며, 거의 동시에 실행이 되는 형태
- → 거의 동시에 실행이 되므로, 실행 순서가 중요한 경우 등의 문제가 있을 경우는 다른 방법이 필요
- 사용 방법은 XML과 동일하며, 내부의 규칙에 맞게 작성하면 된다.
-

# 03. How to use roslaunch

- 가장 기본은
- <launch> 태그를 이용하여, launch 파일임을 선언하는 것으로 부터 시작
- <launch>
- <node … />
- </launch>

- 위와 같은 형태로 작성하여, launch태그 사이에 들어간 내용이 실행될 수 있도록 설정한다.

# 03. How to use roslaunch

- launch파일을 실행하는 terminal에 적용되어있는 terminal 변수를 사용하고 싶은 경우
- $(env variables) 형태로 사용할 수 있다.
- terminal 변수의 경우, terminal에서 $ export variables=50 과 같은 형태로 생성할 수 있다
- 확인의 경우 echo $variables 를 통해 출력해서 확인 가능하다.

```
wego/~/ export variables=50
wego/~/ echo $variables
50
```

```
<param name="foo" value="$(env HELLO)" />
<param name="bar" value="$(optenv WORLD 50)" />
<param name="fooo" value="$(optenv JESUS)" />
```

# 03. How to use roslaunch

- $(env 터미널변수) 방식의 경우, 해당 터미널 변수가 없을 경우, 아예 실행이 되지않는다.
- $(optenv 터미널변수 default값) 형태로 사용하면, 터미널변수가 있을 경우, 그 값이 사용되며, 없을 경우 default 값이 들어간다. 만약 default값이 없을 경우, 비어있는 문자열이 들어가게 된다.

```
wego/~/ export variables=50
wego/~/ echo $variables
50
```

```
<param name="foo" value="$(env HELLO)" />
<param name="bar" value="$(optenv WORLD 50)" />
<param name="fooo" value="$(optenv JESUS)" />
```

- $(find package_name)의 형태를 사용할 경우, 해당 Package까지의 경로를 반환해준다.
- ROS에서는 동일한 이름의 Node를 다수 생성하는 것이 불가능하며, 이 경우 먼저 생성되어 있던 Node가 죽게 되는 현상이 발생한다.
- 이 경우, name부분에 $(anon name) 형태를 사용하면, anonymous id를 사용할 수 있으며, 다수 실행이 가능해진다.

# 03. How to use roslaunch

- <arg name=”this” default=”two” />
- <arg name=”this” value=”one” />
- 위와 같은 형태로, launch파일 내에서 사용할 수 있는 argument를 만들 수 있으며, default의 경우, 뒷부분 또는 launch파일 실행 시 변경이 가능하며, value로 되어있을 경우, 변경이 불가능하다.
- $(arg this) 와 같은 형태로 호출해서 사용할 수 있다.
-

# 03. How to use roslaunch

- 모든 사용 가능한 태그에는 if와 unless에 해당하는 태그가 있으며, if=value 형태로 사용하며, "1" 또는 “True”일 경우, 동작하고 "0" 또는 "False”일 경우 동작하지 않게 한다.
- unless의 경우 if와 반대로 동작한다.
- <param if=”True” name=”hello” value=”world” />
- 위와 같은 형태로 사용할 수 있다.

# 03. How to use roslaunch

- <node> 태그를 사용하여, 가장 기초적인 node를 실행할 수 있다.
- 만약 rosrun hello_pkg hello 라는 이름의 node를 실행하는 것을 launch파일로 작성한다면

- <launch>
- <node name='hello_world' pkg="hello_pkg" type="hello" />
- </launch>

- name의 경우 rosnode list에서 확인할 수 있는 이름을 지정해주면 된다.
- pkg의 경우 실제 실행하려는 node가 있는 패키지의 이름을 넘겨주면 된다.
- type의 경우 실제 실행하려는 node의 이름을 넘겨준다. (cpp일 경우 확장자가 없으며, 파이썬인 경우 .py)

# 03. How to use roslaunch

- 실행 시에 추가적인 argument를 전달하는 것도 가능하며, 이 경우 args=”--test” 와 같은 형태로 속성을 추가할 수 있다.
- respawn=”true” 와 같은 속성을 추가할 경우, node가 특정 문제로 인해 죽은 경우, 다시 실행하도록 할 수 있다.
- respawn_delay=”30” 과 같은 속성을 통해 respawn 사이의 delay를 초 단위로 전달할 수 있다.
- required=”true” 와 같은 속성을 이용하여, 특정 node가 종료되었을 때 전체 launch파일이 모두 종료되도록 할 수 있다.

# 03. How to use roslaunch

- ns=”foo” 와 같은 속성을 통해서 node 및 topic 들 앞에 namespace를 추가할 수 있다.
- clear_params=”true|false” 와 같은 속성을 통해 실행 전에 node에 해당하는 private parameter를 삭제할 수 있다.
- output=”log|screen” 속성을 통해, log를 screen을 통해 확인할 수 있게 할지, 아니면 log에 저장되도록 할지 확인하는 부분을 설정할 수 있다.
- launch-prefix=”prefix arguments” 속성을 통해, gdb, xterm 등을 먼저 실행하고 node를 실행하도록 할 수 있다.

# 03. How to use roslaunch

- <include> 태그를 활용하면, 다른 XML파일 또는 launch파일을 포함시킬 수 있다.
- file="$(find pkg)/path/filename.xml" 과 같은 속성을 추가하면, 다른 xml파일을 추가할 수 있다.
- ns=”foo” 와 같은 속성을 추가하여, namespace를 추가할 수 있다.
- clear_parms=”true|false” 의 속성을 추가하여, 실행 시, 해당 private parameter를 지울 수 있다.
- pass_all_args=”true|false” 의 속성을 추가하여, 현재 launch파일에서 생성한 argument를 포함된 자식 launch파일에도 전달해서 실행할 수 있다.

# 03. How to use roslaunch

- <remap> 태그를 이용하여, node 실행 시 생성되는 Topic의 이름을 변경할 수 있다.
- <remap from=”original_name” to=”new_name” /> 형태로 사용하며, 해당 node 내부에 있는 original_name으로 된 topic을 new_name으로 변경하여 실행한다.
- Subscribe 또는 Publish하는 Topic의 이름을 변경할 수 있다.

# 03. How to use roslaunch

- <param> 태그를 사용하여, 실행할 때, parameter server에 등록할 parameter를 정할 수 있다.
- <param name=”today” value=”20210419” type=”int”/>
- type은 타입이 확실할 경우에 명시하며, 애매할 경우 생략 가능하다.
- name으로 parameter server에 등록이 되며, value 값으로 등록이 된다.
- launch파일 실행 후, rosparam list, rosparam get /today 와 같은 명령어로 확인이 가능하다.
- parameter를 등록 시에, value 부분에서 $(eval 2.* 3.1415 * arg(‘radius’)) 와 같은 형태로 계산된 결과를 입력하는 것도 가능하다.

# 03. How to use roslaunch

- <rosparam> 태그를 이용하여, yaml파일을 불러오거나, 저장할 수 있다.
- <rosparam command=”load” file=”$(find rosparam)/example.yaml” />
- command 속성은 load, dump, delete 세 가지가 있으며, 불러오기, 저장하기, 지우기 세가지이다.
- file은 불러오기 또는 저장하기를 할 때, 경로를 지정해주면 된다.
-

# 03. How to use roslaunch

- <group> 태그를 이용하여, 여러 node들을 한번에 관리할 수 있다
- ns 속성을 통해 그룹의 namespace를 추가할 수 있다.
- clear_params 속성을 통해 실행 시, group namespace의 parameter를 모두 지울 수 있다.
-

# 03. How to use roslaunch

```
<launch>
  <!-- local machine already has a definition by default.
       This tag overrides the default definition with
       specific ROS_ROOT and ROS_PACKAGE_PATH values -->
  <machine name="local_alt" address="localhost" default="true" ros-root="/u/user/ros/ros/" ros-package-path="/u/user/ros/ros-pkg" />
  <!-- a basic listener node -->
  <node name="listener-1" pkg="rospy_tutorials" type="listener" />
  <!-- pass args to the listener node -->
  <node name="listener-2" pkg="rospy_tutorials" type="listener" args="-foo arg2" />
  <!-- a respawn-able listener node -->
  <node name="listener-3" pkg="rospy_tutorials" type="listener" respawn="true" />
  <!-- start listener node in the 'wg1' namespace -->
  <node ns="wg1" name="listener-wg1" pkg="rospy_tutorials" type="listener" respawn="true" />
  <!-- start a group of nodes in the 'wg2' namespace -->
  <group ns="wg2">
    <!-- remap applies to all future statements in this scope. -->
    <remap from="chatter" to="hello"/>
    <node pkg="rospy_tutorials" type="listener" name="listener" args="--test" respawn="true" />
    <node pkg="rospy_tutorials" type="talker" name="talker">
      <!-- set a private parameter for the node -->
      <param name="talker_1_param" value="a value" />
      <!-- nodes can have their own remap args -->
      <remap from="chatter" to="hello-1"/>
      <!-- you can set environment variables for a node -->
      <env name="ENV_EXAMPLE" value="some value" />
    </node>
  </group>
</launch>
```

# 04

timed_roslaunch

# 04. timed_roslaunch

- launch파일의 경우, 실행할 때, 거의 동시에 모든 node가 실행되는 특징이 있다.
- 하지만 상황에 따라 a → b → c와 같은 순서로 node가 실행되어야만 정상적으로 실행되는 경우도 존재
- 이 경우에 사용하는 package가 timed_roslaunch 패키지이다.
- https://github.com/MoriKen254/timed_roslaunch
-

# 04. timed_roslaunch

- 사용 방법은 rosrun timed_roslaunch timed_roslaunch.sh [number of seconds to delay] [rospkg] [roslaunch file] [arguments(optional)] 과 같은 형태로 사용할 수 있다.
- 또한, roslaunch timed_roslaunch timed_roslaunch.launch time:=[number of seconds to delay] pkg:=[rospkg] file:=[roslaunch file] value:=[arguments (optional)] 로 동일하게 사용할 수 있다.
- rosrun timed_roslaunch timed_roslaunch.sh 2 turtlebot_navigation amcl_demo.launch initial_pose_x:=17.0 initial_pose_y:=17.0
-

# 04. timed_roslaunch

- launch파일로 만들어서 실행할 경우도 유사하게 만들 수 있으며

```
<launch>
  <include file="$(find timed_roslaunch)/launch/timed_roslaunch.launch">
    <arg name="time" value="2" />
    <arg name="pkg" value="turtlebot_navigation" />
    <arg name="file" value="amcl_demo.launch" />
    <arg name="value" value="initial_pose_x:=17.0 initial_pose_y:=17.0"/>
    <arg name="node_name" value="timed_roslaunch" /> <!-- This is optional argument -->
  </include>
</launch>
```

- 위 방식 또는 아래 방식을 이용해서 만들 수 있다.

```
<launch>
  <node pkg="timed_roslaunch" type="timed_roslaunch.sh"
    args="2 turtlebot_navigation amcl_demo.launch initial_pose_x:=17.0 initial_pose_y:=17.0"
    name="timed_roslaunch" output="screen" />
</launch>
```

WeGo

# Q & A

go.support@wego-robotics.com

go.sales@wego-robotics.com